DESIGN AND DEVELOPMENT OF AN INTERACTIVE MULTIMEDIA SIMULATION FOR AUGMENTING THE TEACHING AND LEARNING OF PROGRAMMING CONCEPTS

Leonah L. Baloyi, Sunday O. Ojo and Etienne A. Van Wyk
Tshwane University of Technology, Pretoria, South Africa

ABSTRACT

Teaching and learning programming has presented many challenges in institutions of higher learning worldwide. Teaching and learning programming require cognitive reasoning, mainly due to the fundamental reality that the underlying concepts are complex and abstract. As a result, many institutions of higher learning are faced with low success rates in programming courses. This paper presents the design and development of an interactive multimedia simulation (IMS) prototype for augmenting the teaching and learning of programming concepts. An initial requirements elicitation was conducted with the purpose of obtaining the perceptions of programming lecturers and students regarding the programming concepts that present teaching and learning difficulties. The results of this requirements elicitation informed the design and development of an IMS prototype. A design-based research methodology was used which resulted in both a practical and theoretical contribution, i.e. a fully-functional IMS and an evaluation framework for the evaluation of such simulations.

KEYWORDS

Programming Concepts, Interactive, Multimedia, Simulation, Design-based Research

1. INTRODUCTION

Programming is a major part of the computer science curriculum in many institutions of higher learning (Ibrahim *et al*., 2010; Law *et al*., 2010). Programming is taught using different programming languages (Maloney *et al*., 2008). In these programming languages, students are taught different concepts that could be used to solve specific problems. With the underlying concepts being complex and abstract, programming is considered as a hard skill to learn (Shehane & Sherman, 2014; Saeli *et al.*, 2011). The complexities of teaching and learning programming concepts lead to lower success rates and high dropout rates in many universities worldwide (Shehane & Sherman, 2014; Han & Beheshti, 2010; Tan *et al*., 2009). Therefore, institutions of higher learning can add a competitive advantage by focusing their educational strategies on emerging technologies such as interactive and visualization tools in order to match their students' needs (De Gloria *et al.*, 2014; Katai & Toth, 2010). Interactive multimedia simulation is one such tool that can be used to augment the teaching and learning.

Design-based research (DBR) has been defined as a series of approaches that can be used to improve the teaching and learning environment with the aim to produce new theories, artefacts and practices (Barab & Squire, 2004). This paper presents the design and development of an interactive multimedia simulation for augmenting the teaching and learning of programming concepts through three DBR cycles. In general, there are many definitions of an interactive multimedia simulation (IMS) depending on the discipline in which it is used. For purposes of this study, an interactive multimedia simulation can be defined as a 3D (three-dimensional) computer program that combines different multimedia elements, in order to simplify and visualize complex and abstract concepts being taught in the classroom, thereby engaging students and providing user control through an interactive interface, and provide immediate feedback (Saw & Butler, 2008; Vaughan, 2006; Alessi & Trollip, 2001).

ISBN: 978-989-8533-71-5 © 2017

2. RELATED WORKS

Meerbaum-salant et al. (2013) conducted a study to investigate the learning of introductory programming concepts through the Scratch visual environment. Scratch is described as “a visual programming environment that is widely used by young people” (Meerbaum-salant *et al.*, 2013: 69). The results showed that the learning of programming concepts through the Scratch visual programming environment have improved the students' cognitive levels of understanding most concepts. In addition, the number of students’ enrolment in programming have increased. Esteves et al. (2009) conducted a study to examine the use of Second Life (SL) for problem-based learning in programming at the University of Trás-os-Montes e Alto Douro (UTAD), Portugal. SL is described as a 3D online virtual world (Esteves *et al*., 2009). The findings showed that SL can benefit the teaching and learning of a programming language for novice students.

Tan et al. (2009) conducted a study to investigate the learning difficulties in programming concepts for undergraduate students. Accordingly, undergraduate students were invited to participate in a web-based questionnaire. An analysis was conducted and the results had shown that many undergraduate students find it difficult to design a computer program that can solve a specific problem. Yuen (2006) conducted a case study to investigate how an interactive simulation can be used to improve the teaching and learning of programming. The findings from the case study revealed that students were encouraged to think and construct their own solutions through the use of an interactive simulation.

Interactive multimedia simulations were successfully utilized in domains such as medicine and engineering. However, an interactive multimedia simulation (IMS) in this study is distinct from other simulation methods as it was explicitly designed and developed to augment the teaching and learning of programming concepts in institutions of higher learning. Moreover, this IMS is presented through a computer and can be used without any internet connection. Also, the IMS is portable to enable students to copy and use it anywhere, anytime.

3. COGNITIVE THEORY OF MULTIMEDIA LEARNING (CTML)

Mayer and Moreno (2003) drew attention to how people learn in a multimedia environment as learning relies greatly on humans’ cognitive system. This relates to how much information people can take and process into the brain without overloading their cognitive system. The works of Mayer and Moreno (2003) discuss three principles of cognitive theory in multimedia learning (CTML), namely: dual channel, limited capacity and active processing. Dual channel has been defined as an information processing system which consists of two channels to process auditory and verbal information separately. In a limited capacity, the capacity of processing auditory or verbal information in human’s memory is limited, though, the authors stated that presenting pictures and words in a human’s memory is unlimited. Active processing refers to the active processing principle that shows that learning “requires substantial cognitive processing in the verbal and visual channels” (Mayer & Monero, 2003:44). Figure 1 presents the CTML diagram that illustrate the five cognitive processes involved in multimedia learning.

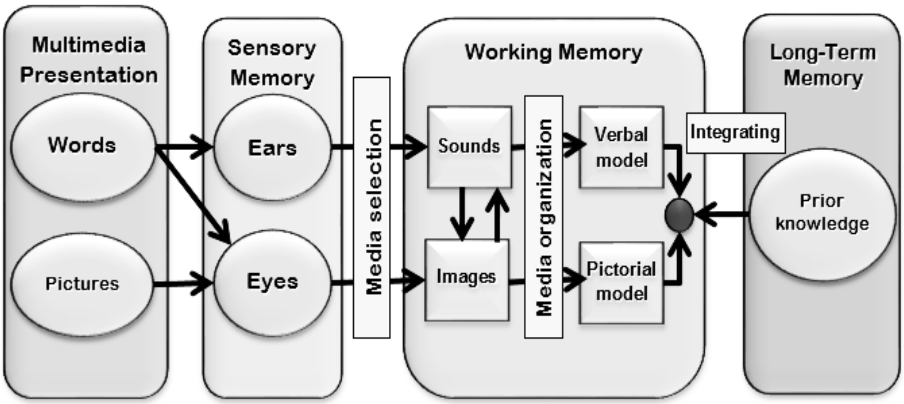

Figure 1. Diagram Presentation of a Cognitive Theory of Multimedia Learning (CTML) Adopted from Mayer (2010:545)

The cognitive processes shown in Figure 1 include the selection of words from a multimedia presentation in working memory, selection of pictures from a multimedia presentation in working memory, organizing the selected words into a verbal representation in working memory, organizing the selected pictures into a pictorial representation in working memory, and integrating the verbal and pictorial representations with previous knowledge activated from long-term memory (Mayer, 2010). Several studies made emphasis that learning involves cognition (Sweller *et al*., 2011; Homer *et al*., 2008; Chipperfield, 2006). Hence, Mayer's CTML guided the design and development of the IMS prototype in this study.

4. RESEARCH METHODOLOGY

The underlying paradigm of this study is design-based research (DBR), due to the evidence in the literature that demonstrates its potential as a methodology suitable to both research and design of technology-enhanced learning environments (Van Wyk & De Villiers, 2014; Amiel & Reeves, 2008; Wang & Hannafin, 2005). Some of the features of DBR as stated in related works are:

Problem-solving: DBR is a problem-solving methodology as it focuses on issues related to practical real-world problems (Van Wyk & De Villiers, 2014; Barab & Squire, 2004).

Appropriate for complex real-world environments: DBR takes place in real-life settings where teaching and learning happen. Therefore, the results obtained from the research provide a sense of validity and can be used to inform and improve educational practice (Barab & Squire, 2004).

Integrative: DBR is integrative as it allows the use of mixed methods in a research to obtain quality and effective results (Van Wyk & De Villiers, 2014).

Iterative: DBR involves multiple iterations. Thus, the process of testing and refinement of prototypes can be repeated in number of cycles until a specific version is accepted (Amiel & Reeves, 2008).

Collaborative and participative: In DBR, the researcher(s) collaborates with participants throughout the analysis and design process to obtain an effective outcome (Amiel & Reeves, 2008; Barab & Squire, 2004).

Dual outcomes: The outcomes of DBR are practical and theoretical. The practical outcomes are in the form of innovative products or interventions while the theoretical outcomes are a sets of design principles or guidelines (Van Wyk & De Villiers, 2014; Barab & Squire, 2004).

This study implemented a mixed methods approach (qualitative and quantitative). These methods include surveys, prototyping and heuristic evaluation. The initial problem that led to this research is low success rates and high drop-out rates in programming courses faced by institutions of higher learning. This study extracted the 2013 to 2016 success rates report for high impact programming subjects at TUT (Tshwane University of Technology), Faculty of Information and Communication Technology (ICT), Computer Science department. The lowest success rate percentage obtained in this report was 14% in Technical Programming IB during 2015 (Tshwane University of Technology, 2017). The low success rates in programming subjects are by no means unique to TUT, but also exist in other institutions of higher learning worldwide (Sarpong *et al*., 2013; Tan *et al*., 2009).

Through the application of DBR in this study, it was significant to initially identify programming problems from a real-world educational setting. Hence, this study conducted requirements elicitation to collect preliminary data. The data for requirements elicitation was collected through survey methods (semi-structured interviews and questionnaires). The researcher conducted semi-structured interviews with twelve programming lecturers. Also, a questionnaire was used to collect data from sixty programming students. The main aim of involving both programming lecturers and students in requirements elicitation was to obtain their perceptions on the difficulties of teaching and learning programming concepts. A further aim was to obtain their views on the proposed solution (IMS) and how it should be designed to alleviate these issues of teaching and learning programming concepts. The findings of the requirements elicitation have shown that both programming lecturers and students perceived three main programming concepts to have

high levels of difficulties. These programming concepts were data types, control structures and array data structures. The outcome of the requirements elicitation directed the study to design and develop the proposed IMS prototype.

5. APPLYING DESIGN-BASED RESEARCH FOR DESIGNING AND DEVELOPING THE IMS

Following the outcome of requirements elicitation, a DBR model was adopted and applied in the design and development of the IMS prototype. DBR has a cyclic approach. The design and development of an IMS prototype was implemented through three DBR cycles, of which each cycle comprised of the five steps of the DBR model. The five iterative steps in each DBR cycle are: problem analysis of a complex problem within real-world, design a solution, development a solution, evaluation in practice and reflection (Van Wyk & De Villiers, 2014). Figure 2 illustrates the DBR model adopted in this study to design, develop and evaluate an IMS prototype.

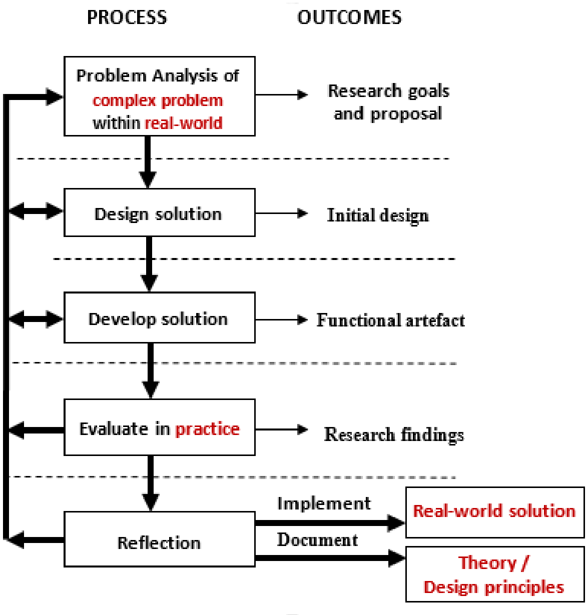

Figure 2. A Generic Model for Design-based Research (DBR), Adopted from Van Wyk and De Villiers (2014)

The explanation of each step of the DBR model is as follows:

1. **Problem analysis -** a practical problem is identified within a context and relevant literature is reviewed to determine the importance of the problem. Problem analysis is conducted to guide the design of a solution.
2. **Design a solution -** an initial design is proposed to address the problem identified in Step 1.
3. **Develop a solution -** A prototype is developed guided by the existing literature.
4. **Evaluate -** a prototype is evaluated to determine its effectiveness.
5. **Reflection -** a researcher reflects on the outcomes of Step1 to 4 which leads to the enhancement of the proposed solution.

To elaborate on how the DBR model was implemented in each DBR cycle for the design and development of an IMS prototype, the following subsections detail the steps of the three DBR cycles.

DBR Cycle 1:

1. Problem analysis

The IMS aimed at addressing the difficulties associated with the teaching and learning of programming concepts. According to the results of requirements elicitation, the abstraction and complexity of programming concepts contribute to the difficulties of teaching and learning. Furthermore, programming lecturers and students suggested that the IMS design should include features such as visualization, real-world examples, interaction and immediate feedback.

2. Design a solution

The findings of requirements elicitation identified three basic programming concepts with high levels of difficulties as indicated by both programming lecturers and students. These three basic programming concepts are data types, control structures and array data structures. As a result, the storyboard scenarios were designed to address the abstraction and complexity associated with teaching and learning of these three basic programming concepts. The outcome of this step informed the development of the solution.

3. Develop a solution

Guided by the literature, a low-fidelity prototype in the form of a storyboard was drawn on paper to demonstrate the scenarios designed in Step 2. Generally, a storyboard illustrates all the items that will be heard, seen or experienced by the user in the system or tool. Accordingly, the storyboard drawings used real-world examples to illustrate the three basic programming concepts (data types, control structures and array data structures). The storyboard comprised of scenes such as the menu interface, introduction video, practical exercises and assessment scene. Figure 3 presents an example of the storyboard scenes.

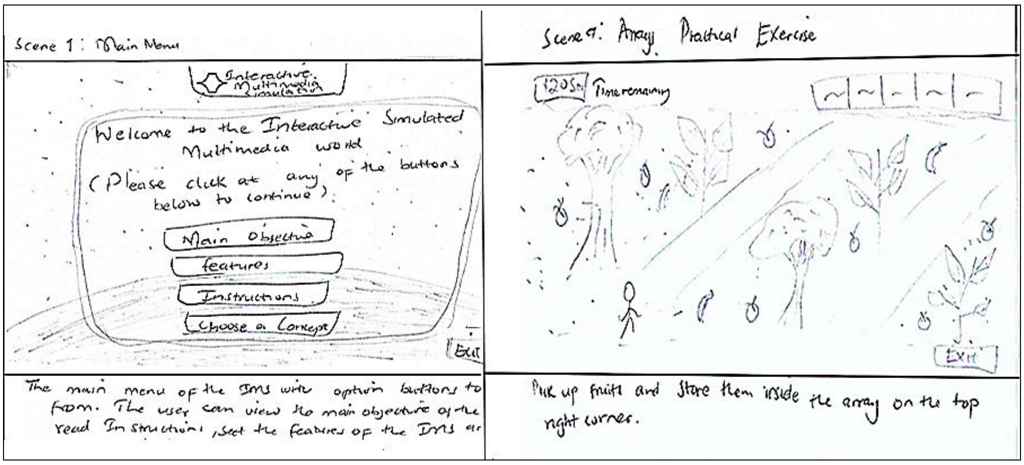

Figure 3. Storyboard Scenes Indicating the Main Menu and Sub Menu of the IMS

4. Evaluate

The storyboard was evaluated by the two research supervisors for its effectiveness. Both supervisors have expertise in computer science education, multimedia design and simulations. The outcome of the evaluation led to the refinements of the storyboard scenarios formulated in Step 2.

5. Reflection

The researcher reflected on the outcomes of Step 1 to 4 which provided the need to do further improvements on the logical progression of scenes in the storyboard. Also, the storyboard scenarios were still inappropriate. The outcome of this step led to a second cycle of problems analysis, designing a solution, developing a solution, evaluation and reflection steps.

DBR Cycle 2:

1. Problem analysis

Certain issues were identified from DBR Cycle 1 which led to this cycle. The issues identified were inappropriate storyboard scenarios and the storyboard scenes were not arranged in sequence. The outcome of problem analysis in this step prompted the design of a solution.

2. Design a solution

Guided by the existing literature, the content in the storyboard scenarios was refined and the storyboard scenes were arranged in sequence. The evaluation framework for evaluating the IMS prototype was also designed. The evaluation framework comprised of three categories, namely: Instructional design, General usability and Interactive multimedia simulation design. Each category had a number of criteria and evaluation statements. This evaluation framework is an additional theoretical output of the overall study, but falls outside the scope of this paper and hence it is not described in detail.

3. Develop a solution

Deriving from the improved storyboard in the previous step, a high-fidelity prototype was developed in the form of a fully-functional interactive multimedia simulation (IMS). In the same way as the storyboard, the IMS prototype comprised of three basic programming concepts (data types, control structures and arrays data structures). The following software and hardware were used in the design and development an IMS prototype:

Software:

- The Autodesk 3Ds Max 15 application was used for modelling 3D graphics and creating animations.
- Unity 3D 4.6.2 Pro version game engine was used for designing and developing the IMS prototype scenes, integrating multimedia elements, scripting and for publishing the IMS prototype.
- Adobe Photoshop CS6 was used for designing the 2D graphics.
- Adobe after effects CS6 was used for designing the introduction videos of each programming concept in the IMS prototype.
- Adobe Audition CS6 was used for audio editing.

Hardware:

- Desktop computer/laptop was a platform used for designing, developing and publishing the IMS prototype.
- A keyboard was used for input and interaction with the IMS prototype environment throughout the design and development process.
- A mouse was used for input and interaction with the IMS prototype environment throughout the design and development process.
- The audio recording equipment was used to record voice narrations.
- Multimedia speakers were used for audio output.

Figure 4 is a screen shot of the IMS prototype scene, it shows a man called Bob sleeping in the bedroom while the clock alarm is ringing to wake him up. This screen shot illustrates how real-life events relate to an IF-statement in control structures programming concept. If the statement shown in the flowchart is true, the user should press “enter button” from the keyboard to wake Bob up.

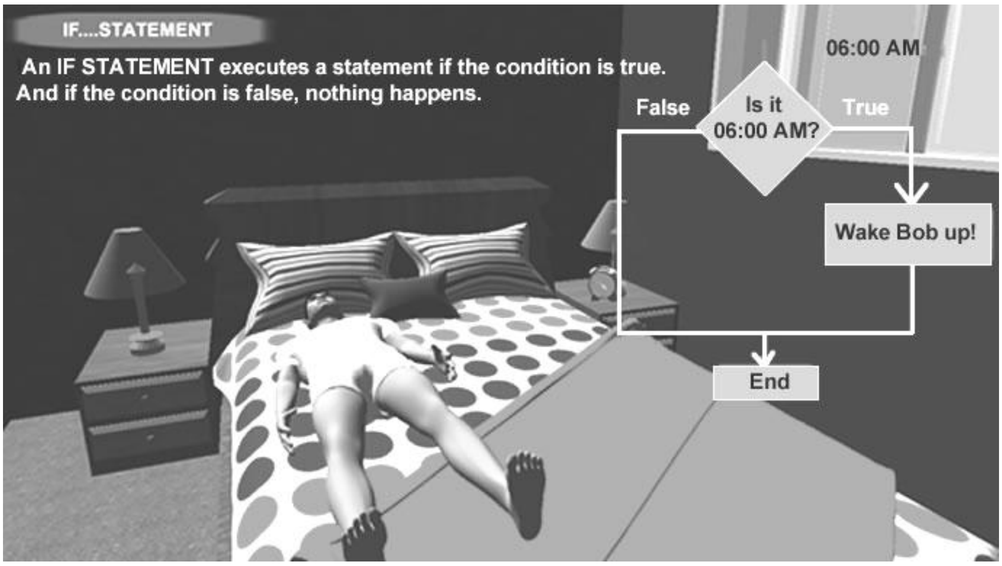

Figure 4. IMS Prototype Scene Showing a Man Sleeping and the Alarm Is Ringing to Wake Him Up

4. Evaluate

Heuristic evaluation was conducted to evaluate the IMS prototype. Heuristic evaluation is an inspection method where-by a small number of experts apply a set of principles to judge aspects of an interface or a specific tool in order to improve its appropriateness (Madan & Dubey, 2012). The evaluation framework designed in Step 2 was used for heuristic evaluation to evaluate the IMS prototype. Heuristic evaluation was performed by nine expert evaluators. The expert evaluators were three instructional designers, three usability specialists and three interactive multimedia simulation designers. Each group of three experts evaluated a specific category on the evaluation framework.

5. Reflection

During heuristic evaluation, the nine expert evaluators identified several design and contextual issues in the IMS prototype. Some of these issues were lack of hints and narrations to guide the user while performing some tasks on the IMS prototype, inappropriate programming content, poor navigation and inappropriate animations. Additionally, a new set of criteria was suggested on the evaluation framework itself. The outcome of this step led to a third cycle of problems analysis, designing a solution, developing a solution, evaluation and reflection.

DBR Cycle 3:

1. Problem analysis

As mentioned on the reflection step of DBR Cycle 2, several design and contextual problems were identified by expert evaluators. In addition, a new set of criteria were proposed on the evaluation framework. Accordingly, literature was reviewed to guide the design of the new set of criteria for the evaluation framework.

2. Design a solution

The new set of criteria such as relevant subject matter, fostering of germane load and artistic suitability were designed and incorporated into the evaluation framework. The new revised evaluation framework comprised of the same three categories as mentioned in DBR Cycle 2 but,

with a new set of additional criteria and associated evaluation statements as recommended by the nine expert evaluators.

3. Develop a solution

The contextual and design issues such as lack of hints and narrations to guide the user while performing tasks, inappropriate programming content, poor navigation and inappropriate animations on the IMS prototype as previously identified by expert evaluators in DBR Cycle 2 were addressed and improved upon. The outcome of this step prompted a second round of heuristic evaluation to ensure the effectiveness of the enhanced IMS prototype and evaluation framework.

4. Evaluate

The second round of heuristic evaluation was conducted using the improved evaluation framework to evaluate the enhanced IMS prototype. The heuristic evaluation involved the same nine expert evaluators from DBR Cycle 2.

5. Reflection

The findings of the second round of heuristic evaluation still indicated some minor design issues on the IMS prototype such as spelling errors, inappropriate text and animations, and low quality graphics. As a result, the researcher refined the IMS prototype to address these design issues. The outcome of this cycle lead to the improved IMS prototype and evaluation framework. By now, the prototype has been refined into a fully-functional IMS.

6. DISCUSSION

It was an aim of this study not just to find a solution to the main problem but also, to document the procedures that were followed to obtain the solution. The three DBR cycles presented in the previous section described the application of the DBR model to design and develop an IMS porotype for augmenting the teaching and learning programming concepts. To indicate how this research adheres to the features of DBR presented in Section 4, these features are revisited with an explanation of how each was applied (see Table 1).

Table 1. The Application of DBR Features in This Research

DBR Features	Application to this research
Problem-solving	Teaching and learning programming require cognitive reasoning, mainly due to the programming concepts being complex and abstract in nature. This research had focused on solving a real-world practical problem by designing and developing an interactive multimedia simulation (IMS) to augment the teaching and learning of programming concepts.
Appropriate for complex real-world environments	The institutions of higher learning are complex environments that are usually faced with similar teaching and learning problems. Therefore, the results obtained from the use of an IMS to augment the teaching and learning of programming concepts in this research can be significant to inform similar studies.
Integrative	This research implemented mixed methodologies such as survey, prototyping and heuristic evaluation in order to obtain effective results.
Iterative	As indicated in the previous section, three cycles of iterative process of analysis, design, development, evaluation and reflection was followed by the researcher to design and develop an IMS.

Collaborative and participative	Initially, the researcher collaborated with programming lecturers and students with the purpose of obtaining their perceptions regarding the programming concepts that present teaching and learning difficulties. Additionally, throughout the IMS design, development and evaluation process, the researcher collaborated with programming lecturers, instructional design experts, usability experts and interactive multimedia design experts.
Dual outcomes	The final output of this study is a fully-functional IMS and an evaluation framework for the evaluation of such simulations.

The evaluation framework was mentioned in this paper due to its function in the study and the fact that the refined framework is the theoretical contribution of the study, but details of the evaluation framework falls outside the scope of this paper.

7. CONCLUSION

This paper presented the design and development of an interactive multimedia simulation (IMS) for augmenting the teaching and learning of programming concepts. A DBR model was adopted from literature and implemented through three DBR cycles to design and develop an IMS prototype. The results of this study are a practical contribution in the form of a fully-functional IMS and a theoretical contribution in the form of an evaluation framework. Therefore, this study recommends the use of DBR in educational research in order to improve both theory and practice.

REFERENCES

Alessi, S. and Trollip, S., 2001. Simulations. *Multimedia for Learning (Methods and Development)*, 213-269.

Amiel, T. and Reeves, T. C., 2008. Design-based research and educational technology: Rethinking technology and the research agenda. *Educational Technology & Society*, 11, 29-40.

Barab, S. and Squire, K., 2004. Design-based research: Putting a stake in the ground. *The journal of the learning sciences*, 13, 1-14.

Chipperfield, B., 2006. Cognitive Load Theory and Instructional Design Saskatoon. Saskatchewan, Canada: University of Saskatchewan (USASK).

Cooper, S. et al, 2000. Alice: a 3-D tool for introductory programming concepts. Journal of Computing Sciences in Colleges, 2000. Consortium for Computing Sciences in Colleges, 107-116.

De Gloria, A. et al, 2014. Serious Games for education and training. *International Journal of Serious Games,* 1.

Esteves, M. et al, 2009. Using Second Life for problem based learning in computer science programming. *Journal for Virtual Worlds Research,* 2.

Han, J. and Beheshti, M., 2010. Enhancement of computer science introductory courses with Mentored Pair Programming. *Journal of Computing Sciences in Colleges,* 25, 149-155.

Homer, B. D. et al, 2008. The effects of video on cognitive load and social presence in multimedia-learning. *Computers in Human Behavior,* 24, 786-797.

Ibrahim, R. et al, 2010 Using educational games in learning introductory programming: A pilot study on students' perceptions. Information Technology (ITSim), 2010 International Symposium in, 2010. IEEE, 1-5.

Katai, Z. and Toth, L., 2010. Technologically and artistically enhanced multi-sensory computer-programming education. *Teaching and teacher education,* 26, 244-251.

Law, K. M. et al, 2010. Learning motivation in e-learning facilitated computer programming courses. *Computers & Education,* 55, 218-228.

Madan, A. and Dubey, S. K., 2012. Usability evaluation methods: a literature review. *International Journal of Engineering Science and Technology,* 4, 590-599.

Maloney, J. H. et al, 2008. *Programming by choice: urban youth learning programming with scratch*, ACM.

Mayer, R. E. and Moreno, R., 2002. Animation as an aid to multimedia learning. *Educational psychology review,* 14, 87-99.

Mayer, R. E. and Moreno, R., 2003. Nine ways to reduce cognitive load in multimedia learning. *Educational psychologist,* 38, 43-52.

Mayer, R. E., 2010. Applying the science of learning to medical education. *Medical education,* 44, 543-549.

Meerbaum-Salant, O. et al, 2013. Learning computer science concepts with scratch. *Computer Science Education,* 23, 239-264.

Saeli, M. et al, 2011. Teaching programming in secondary school: a pedagogical content knowledge perspective. *Informatics in Education-An International Journal*, 73-88.

Sarpong, K. A. M. et al, 2013. Causes of Failure of Students in Computer Programming Courses: The Teacher-Learner Perspective. *International Journal of Computer Applications,* 77.

Saw, J. L. Y. and Butler, M., 2008. *Exploring graphical user interfaces and interaction strategies in simulations*, Monash University.

Shehane, R. and Sherman, S., 2014. Visual teaching model for introducing programming languages. *Journal of Instructional Pedagogies,* 14, 1.

Sweller, J. et al, 2011. Measuring cognitive load. *Cognitive load theory.* Springer.

Tan, P. H. et al, 2009. Learning difficulties in programming courses: Undergraduates' perspective and perception. Computer Technology and Development, 2009. ICCTD'09. International Conference on, 2009. IEEE, 42-46.

Tshwane University Of Technology, 2017. *Success Rates of programming subjects - Department of Computer Science.*

Van Wyk, E. and De Villiers, M., 2014. Applying design-based research for developing virtual reality training in the South African mining industry. Proceedings of the Southern African Institute for Computer Scientist and Information Technologists Annual Conference 2014 on SAICSIT 2014 Empowered by Technology, 2014. ACM, 70.

Vaughan, T., 2006. *Multimedia: Making it work*, Tata McGraw-Hill Education.

Yuen, A. H., 2006. Learning to program through interactive simulation. *Educational Media International,* 43, 251-268.